#11 Reviewed 5/13/03

RECEIVED
MAY 0 8 2003
Technology Center 2100

CERTIFICATE OF TRANSMISSION
37 C.F.R 1.8

I hereby certify that this correspondence is being facsimile transmitted to the United States Patent and Trademark Office

Fax No. (703) 746-7239 on 8 May 2003 (Date)

Typed or printed name of person signing this certificate:

RICHARD C. AUCHTERLONIE, REG. 30,607

8 May 2003 (Date) Richard C. Auchterlonie (Signature)

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:
Uresh K. Vahalia & Percy Tzelnic

Serial No.: 09/261,621

Filed: March 3, 1999

For: File Server System Providing Direct Data Sharing Between Clients With A Server Acting As An Arbiter and Coordinator

Group Art Unit: 2154

Examiner: Dustin Nguyen

Atty. Dkt. No.: EMCR:039/AUC

REPLY TO OFFICIAL ACTION DATED MARCH 17, 2003

Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Sir:

In reply to the Official Action dated March 17, 2003, please note that the specification was amended on page 3 line 4 and page 13 lines 12-13 by a Preliminary Amendment filed Sep. 6, 1991, to indicate that Vahalia et al, Serial No. 08/747,631 filed Nov. 13, 1996, issued as U.S. Patent 5,893,140 on Apr. 6, 1999. Formal drawings were also filed on Sep. 6, 1991.

He meant → 4/6/01

H: 52872(BBY_011.DOC)

The present invention relates generally to data storage systems, and more particularly to network file servers. (Applicants' specification, page 1, lines 2-4). In particular, data consistency problems may arise if concurrent client access to a read/write file is permitted through more than one data mover. (Applicant's specification, page 3, lines 1-2.)

The present invention provides a method of operating a file server in a data network. The file server receives a request for metadata about a file to be accessed. The request is received from a data processing device in the data network. In response to the request for metadata, the file server grants to the data processing device a lock on at least a portion of the file, and returns to the data processing device metadata of the file including information specifying data storage locations in the file server for storing data of the file. (Claim 1.)

For example, as shown in applicants' FIG. 3, before reading or writing to the file system 62, a client first issues a request for metadata to the data mover 61. The data mover 61 responds by placing an appropriate lock on the file to be accessed, and returning metadata including pointers to where the data to be accessed is stored in the file system. The client uses the metadata to formulate a read or write request sent over the bypass data path to the file system 62. If the write request changes the file attributes, then the client writes the new file attributes to the data mover 61 after the data is written to the file system 62. (Applicants' specification, page 14, lines 18-25.)

In another example, shown in FIG. 4, a second client 88 accesses the first file system 83 in the fashion described above with reference to FIG. 3, and a third client 89 accesses the second file system 84 in the fashion described above with reference to FIG. 3. For example, to access the first file system 83, the second client 88 sends a metadata request to the first data mover 81.

The first data mover 81 places a lock on the file to be accessed, and returns metadata including pointers to the data in the file to be accessed. The second client 88 uses the pointers to formulate a corresponding data access command sent over the bypass data path 92 to the first file system 83, and any read or write data is also communicated over the bypass data path 92 between the first file system 83 and the second client 88. (Applicants' specification, page 16, lines 7-15.)

The file server 60 in FIG. 3 provides direct data sharing between network clients 64, 65 by arbitrating and coordinating data access requests. The data mover 61 grants file lock request from the clients 64, 65 and also provides metadata to the clients 64, 65 so that the clients can access data storage 62 in the cached disk array 63 over a data path that bypasses the data mover 61. The data mover 81, 82 and the clients 88, 89 in FIG. 4 may operate in a similar fashion. (Applicant's specification, page 55, lines 18-24.)

The network file server architecture of FIG. 4 allows file sharing among heterogeneous clients, and supports multiple file access protocols concurrently. The architecture permits clients using traditional file access protocols to inter-operate with clients using the new distributed locking and metadata management protocol for direct data access at the channel speed of the data storage devices. This provides a scaleable solution for full file system functionality for coexisting large and small files. (Applicants' specification, page 58, line 30, to page 59, line 5.)

On page 2 of the Official Action, claims 1-8, 11-20, 27, 30-35, and 42-50 were rejected under 35 U.S.C. 103(a) as being unpatentable over Burns et al. (U.S. Patent No. 6,088,694, hereinafter Burns) in view of Bennett et al. (U.S. Patent No. 5,852,747, hereinafter Bennett). Applicants respectfully traverse.

The policy of the Patent and Trademark Office has been to follow in each and every case the standard of patentability enunciated by the Supreme Court in Graham v. John Deere Co., 148 U.S.P.Q. 459 (1966). M.P.E.P. § 2141. As stated by the Supreme Court:

> Under § 103, the scope and content of the prior art are to be determined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background, the obviousness or nonobviousness of the subject matter is determined. Such secondary considerations as commercial success, long felt but unsolved needs, failure of others, etc., might be utilized to give light to the circumstances surrounding the origin of the subject matter sought to be patented. As indicia of obviousness or nonobviousness, these inquiries may have relevancy.

148 U.S.P.Q. at 467.

The problem that the inventor is trying to solve must be considered in determining whether or not the invention would have been obvious. The invention as a whole embraces the structure, properties and problems it solves. In re Wright, 848 F.2d 1216, 1219, 6 U.S.P.Q.2d 1959, 1961 (Fed. Cir. 1988).

Burns discloses a computing system for providing continuous availability and efficient backup for externally referenced objects. (Title.) The system is said to provide continuous availability of data files that are maintained at a file management system and linked to a database management system (DBMS) through a Datalink data type, even while any particular file is being changed with DBMS append or update operations. When a file is linked, it is designated to be available for read-only operations. A user who wants to perform updates on a file gets a "check-out" copy of the file for updating operations, such that the original file remains linked to the database system while the copy is being updated and remains available to other users. The file

management system includes a "check-in" function that receives the updated file, saves the updated file under a new name different from the original, updates the Datalink, generates new metadata for the updated file, and transactionally updates the file with its new metadata. In this way, data files are continuously available to all users through appending and updating actions. Since updating Datalinks requires the file management system to initiate backup, a "delta versioning" operation, which reduces the data needed to support backup operations, permits more efficient backup of data files and enables the continuously available files to be backed up and consistent. (Abstract.)

With reference to claim 1, Burns discloses a file server (17 in Burns FIG. 1). The file server responds to requests from a user's computer such as requests for retrieval, updating, and deletion of files, and addition of new data to the data base. The computing system user can communicate with the file manager 18 through the file system API 28 over a file communication path 34 to create, store, and request files in the file system. (Burns, col. 7, lines 33-43.) The computing system of Burns also includes a database management system (DBMS, 15 in FIG. 1). In general, a DBMS associates its data records with metadata that includes information about the storage location of a record, the configuration of the data in the record, and the contents of the record. (Burns, col. 1, lines 55-59.) The DBMS incorporates metadata for each file in the underlying file management system and uses the metadata to more easily retrieve a particular file, keep track of file versions, and retrieves files based on contents. (Burns, col. 2, lines 36-39.) The DBMS works with interface programs such as the "DataLinks" application to provide a general-purpose query engine to manage file access, referential integrity, and indices. This is said to permit arbitrary large data objects to be associated with the DBMS while preserving the file system interface. In this way, existing file

management system applications using the Datalink data type can access the data directly without import and export of data through the database system. (Burns, col. 3, lines 8-17.) For example, when the application or user terminates the append function, the database management system transactionally updates the referring Datalink column at the DBMS to link the file in its modified/append state. (Burns, col. 4 line 67 to col. 5 line 4.) The Datalink column 63 in the DBMS includes "serverx/filename" references. (Burns, FIG. 3, col. 8, lines 58-63.)

Bennett discloses a client/server computer system that manages shared files. A client includes a data cache and an associated cache manager, and executes a client application that requests data from a shared file. If the data is not currently stored in the cache, the client cache manager sends to the server a request for multiple consecutive blocks of data beginning with the first block containing the data requested by the client from the shared file. The server includes a token manager which receives the request, and in response (a) awards the token for a first data block specified in the request regardless of contention for the first data block and (b) awards tokens for multiple blocks held by a client who also holds the token for the first block. (Bennett, Abstract.)

With reference to applicants' claim 1, Bennett discloses a method of operating a file system in a data network, the method comprising; (a) the file server receiving a request for metadata about a file to be accessed, the request being received from a data processing device (i.e., a client) in the data network (Bennett, column 4, lines 4-7 and 23-28); and (b) in response to the request for metadata, the file server granting to the data processing device a lock (i.e., a token) on at least a portion of the file (at least the first data block; see Bennett, Abstract, lines 11-

22) and returning to the data processing device metadata of the file (e.g., Bennett, column 4, lines 25-27).

With respect to the differences between the applicants' claim 1 and the cited art, Burns fails to disclose the file server "returning to the data processing device metadata of the file including information specifying data storage locations in the file server for storing data of the file." In Burns, if a user computer does not know what file or file version to access, the user computer can access the DBMS to obtain one or more "serverx/filename" references, each of which identifies a file server and file name of a file in the file server. However, a "serverx/filename" reference does not specify data storage locations in the file server for storing data of the file. In Burns, the file server 17 keeps track of what data storage locations in the file storage 19 are associated with any particular file, and there is nothing in Burns to suggest that the file server 17 specifies data storage locations to either the document management system 15 or the user computer 22. Instead, to access data of a specified file in the file server 19, the user computer 22 sends a conventional file access request to the file sever 17, and the file server 17 interprets the file access request to access the file storage 19. (See Burns col. 7 lines 15-17, 25-28, and 36-43.)

In contrast, the applicants' specification describes the applicants' file server providing a user with "pointers to where the data to be accessed is stored in the file system." (Page 14, lines 22-23.) This permits the client to use the pointers to "formulate a read or write request sent over the bypass data path to the file system 62." (Page 14, lines 22-23). For example, the pointers point "to where the file data resides in the cached disk array storing the file system." (Page 31 lines 13-14). Moreover, a file name does not specify data storage locations in the file server for

storing data of the file because a primary function of the file manager is to map file names to data storage locations allocated to the files by the file manager. (See applicants' specification, page 2, lines 17-20; Burns, col. 7, lines 15-17.)

Page 3, paragraph 5 of the Official Action recognizes that "Burns does not disclose in response to the request for metadata, the file server granting to the data processing device a lock on at least a portion of the file, and returning to the data processing device metadata of the file."

The previous Official Action dated Sep. 16, 2002, page 3, paragraph 4, recognized: "Bennett does not disclose the metadata of the file including information specifying data storage locations in the file server for storing data of the file." Instead, Bennett returns the file identification, file type, file size, date of last change, etc. (Bennett, column 4, lines 25-27).

On page 3, paragraph 5, the Official Action concludes: "It would have been obvious to a person skill[ed] in the art to combine the teaching of Burns and Bennett because Bennett's granting a lock would preserve the integrity of the file by granting only lock to portion of the file to prevent unauthorized access." The applicants respectfully disagree. It is not seen how any proper combination of Burns and Bennett would result in applicants' invention of claim 1. As discussed above, neither Burns nor Bennett discloses that information specifying data storage locations in the file server for storing data of the file should be returned by the file server to a data processing device in the data network in response to a request from the data processing device to the file server for metadata about the file. Nor would there be any reason for returning information specifying data storage locations in the file server for storing data of the file if the file server 10 of FIG. 1 of Bennett were substituted for the file server 14 in the system of Burns. Such a suggestion in the Official Action is improper hindsight unless it is found in the prior art as

a whole apart from the teaching in applicants' novel disclosure. Moreover, the applicants' specification, page 17, lines 1-10, teach that use of the invention is desirable only under particular circumstances:

> Whenever a client has a bypass data path to a file system and can therefore send data access commands to the file system without passing through a data mover computer, the client can potentially access all of the files in the file system. In this situation, the client must be trusted to access only the data in a file over which the client has been granted a lock by the data mover that owns the file system to be accessed. Therefore, the methods of client access as described above with reference to FIGS. 2 and 3 have a security risk that may not be acceptable for clients located in relatively open regions of the data network. The method of client access as described above with reference to FIG. 3 also requires special client software, in contrast to the methods of client access as described above with reference to FIGS. 1-2 which can use standard client software.

Where the prior art references fail to teach a claim limitation, there must be "concrete evidence" in the record to support an obviousness rejection. "Basic knowledge" or "common sense" is insufficient. *In re Zurko*, 258 F.3d 1379, 1385-86, 59 U.S.P.Q.2d 1693, 1697 (Fed. Cir. 2001).

With respect to claim 2, page 4, paragraph 6 of the Official Action cites Burns col. 1, lines 44-59, but it is not seen where this passage discloses the data mover computer maintains a metadata cache in the random access memory.

With respect to claim 4, pages 4-5, paragraph 8 of the Official Action cites Bennett e.g. column 4, line 23-38 for a disclosure that the data processing device writes data to the data storage locations in the file server, modifies (updates) metadata from the file server in accordance with the data storage locations in the file server to which the data is written, and

sends the modified metadata to the file server. However, Bennett does not disclose that the data processing device (client) writes data to the data storage locations in the file server. Instead, Bennett's file server 16 writes data to data storage locations in the DASD 61. Hence, the data processing device (client) of Bennett is not modifying the metadata from the file server "in accordance with the data storage locations in the file server to which the data is written."

In a conventional file server, it is the file server that is mapping the file name to the storage locations, and modifying the metadata of the file when needed in accordance with the data storage locations in the file server to which the data is written. Claim 4 instead defines that the data processing device is writing data to the data storage locations in the file server, and modifying the metadata from the file server in accordance with the data storage locations in the file server to which the data is written. In short, claim 4 defines that the data processing device (e.g., the client) is performing operations previously performed by the file server.

With respect to claim 5, the Official Action, page 5, paragraph 9, recognizes that Burns does not disclose the data processing device sends the modified metadata to the file server after the data processing device writes the data to the data storage of the file server. As discussed above with respect to claim 4, the data processing device (client) of Bennett does not write the data to the data storage of the file server; instead, Bennett's file server 16 writes data to data storage locations in the DASD 6.

With reference to claim 6, Burns has different metadata about different versions of a file. Claim 6, however, is dealing with a different problem of different versions of metadata of the same file, and in particular the file server responding to a file access request from the data processing device by returning new metadata to the data processing device only when the data

processing device has a cache of obsolete metadata of the file. Therefore the Burns method of maintaining a database of metadata about different versions of files fails to suggest the particular recitations in claim 6. For example, it is not seen where the file server of Burns compares a version identifier from the data processing device to a version identifier of a most recent version of the metadata of the file, and the file server returns the most recent version of the metadata of the file to the data processing device when the comparison of the version identifier from the data processing device to the version identifier of the most recent version of the metadata of the file indicates that the metadata of the file cached in the cache memory of the data processing device is not the most recent metadata of the file.

With reference to claim 8, Burns and Bennett have been distinguished above with reference to the recitations in claim 8 that are similar to the recitations in claims 1 and 4. In particular, it is not seen where Burns or Bennett discloses the file server sending to the client "information specifying data storage locations in the file server for storing data of the file", nor the client producing a data access command from the metadata for accessing these data storage locations in the file server. Moreover, the cited passages of Burns [col. 9, lines 1-6 and line 59 to col. 10, line 4] relate to the user's computer sending an SQL command to the database management system to access a column of type Datalink, which issues a "LinkFile" or "UnlinkFile" command to the file server, causing the database file server to recognize the database management system 15 as the owner of the file in the file system, thereby preventing any file system user from renaming or moving the file. In effect, the user's computer is using the database management system to locate a file server and a file in the file server and to send a command to the file server to place a lock on the file. It is not understood how this would

suggest the applicant's method of claim 8 which defines that the file server responds to a request for access to a file by placing a lock on the file and sending to the client metadata of the file including information specifying data storage locations in the file server for string data of the file, and the client using this metadata of the file for producing at least one data access command for accessing the data storage locations in the file server, and sending the data access command to the file server to access the data storage locations of the file server. The method of Burns is different from the claimed method, and there is nothing to suggest that the method of Burns is in any way deficient for carrying out its intended purpose of locating a particular file to access and preparing that file for access.

With reference to claim 12, in the cited passage in Burns, col. 9 lines 8 to 15, say:

> Such constraints, include, for example, making a database system the owner of the named file and marking the file as a read-only file. This linkage is provided in a transactional manner. The rationale for changing the owner of the file to a database system from the file system user is to prevent the file from being renamed or deleted by the file system users, which guarantees the integrity of any reference made in the database system to the file.

This passage appears to teach away from granting a lock to the client. Nor is it seen where the File System API 28 is a lock manager that grants a local lock to a particular application process that accesses the file. Instead, one might expect that the file manager 18 of the file server 14 would have a conventional lock manager that would grant file locks to particular client processes; for example, the access control in FIG. 5 of Burns could include a client process identifier.

With reference to claim 13, page 7 paragraph 15 of the Official Action cites Burns et al., col. 9, lines 22-38, which say:

> With reference to FIGS. 1 and 3, the system 10 includes an application such as the application 24, one or more standard interfaces such as an SQL API 27 for database access, and a file system API 28 for standard file system calls (such as open, read, close) to access files. An application scenario can be understood as follows: Assume the application 24 issues an SQL "SELECT" statement to search on the database in the database storage 16. Assume further that the database includes the relation defined by the data table 60 (FIG. 3). In this regard, the query returns its results, which include one or more "serverx/filename" references, as column data in the Datalink data structure (assuming any Datalink column is selected in the query). The application 24 can then use the file system API 28 and the file communication path 34, employing standard file system protocols to access the relevant portion of a file.

It is not understood how this passage of Burns suggests that the client has a lock manager. Instead, the passage describes a client using the database management system to search for a particular file and obtain a "servx/filename" reference, and then access the file server and filename specified by the "servx/filename" reference. Nor is it seen where the File System API 28 is a lock manager that grants a local lock to a particular application process that accesses the file. Instead, one might expect that the file manager 18 of the file server 14 would have a conventional lock manager that would grant file locks to particular client processes; for example, the access control in FIG. 5 of Burns could include a client process identifier.

With reference to claims 15 and 16, these claims include recitations similar to the recitations of claims 4 and 5, and therefore Burns and Bennett are distinguished for the reasons discussed above with respect to claims 4 and 5.

With reference to claim 17, page 7, paragraph 18 of the Official Action cites Burns, col. 11, line 64 to column 12, line 10. This refers to a file owner (user) completing an append operation, and then initiating an update operation at the database management system. The update operation is said to be equivalent to an atomic unlink-relink process and results in an

asynchronous backup operation being performed by the file manager. In other words, in Burns, when a client is finished updating a new version of a file, the client accesses the database management system to cause the new version of the file to be substituted for the old version of the file, and to cause the file server to backup the new version. This is different from the applicants' method of claim 17 where the client performs asynchronous write operations upon the data storage locations of the file server (i.e., without keeping the metadata of the file current), and then sends the modified metadata to the file server in response to a commit request from the application process of the client. In particular, in Burns, it appears that the client performs the append operation upon the new version of the file by sending conventional file access commands to the file server 17, which would not be an asynchronous write operation because the file server 17 would keep the metadata of the new version of the file current. Nor is it seen where the client in Burns sends modified metadata of the file to the file server 17, because the file server would itself keep the metadata of the new version of the file current.

With reference to claim 18, page 8, paragraph 19 of the Official Action cites Burns col. 4, lines 43-46, which says:

> When a modifying application finishes writing the data on the local file system, it closes its write access and transactionally updates the Datalink reference to the linked file, to reference the changed file data.

However, this is different from the applicants' method of claim 18, in which the client performs asynchronous write operations upon the data storage locations of the file server, and wherein the client sends the modified metadata to the file server when the client requests the file server to close the file. In Burns, it appears that the client performs the write operation upon the new version of the file in the local file system by sending conventional file access commands to

the file server 17, which would not be an asynchronous write operation because the file server 17 would keep the metadata of the new version of the file current. Nor is it seen where the client in Burns sends modified metadata of the file to the file server 17, because the file server would itself keep the metadata of the new version of the file current.

With reference to claim 19, see the remarks above with respect to claim 6.

With reference to claim 27, see the remarks above with respect to claim 8. In particular, neither Burns nor Bennett discloses a file server programmed for responding to a request for access to a file from a client by granting to the client a lock on the file and sending to the client metadata of the file including information specifying data storage locations in the file server for storing data of the file.

With reference to claim 31, see the remarks above with respect to claim 12.

With reference to claim 32, see the remarks above with respect to claim 13.

With reference to claim 34, see the remarks above with respect to claim 15.

With reference to claim 42, see the remarks above with respect to claim 8.

With reference to claim 43, see the remarks above with respect to claim 13.

With reference to claims 45 to 49, see the remarks above with respect to claims 15 to 19, respectively.

With reference to claim 50, see the remarks above with respect to claims 1, 2, 4, and 5. In addition, claim 50 defines that the client sends the data access command (specifying the data storage locations in the cached disk array for storing the data to be written) over a data path that bypasses the data mover computer to access the data storage locations in the cached disk array for storing the data to be written.

On page 9, paragraph 28 of the Official Action, claims 9, 10, 21-26, 28, 29, and 36-41 were rejected under 35 U.S.C. 103(a) as being unpatentable over Burns in view of Bennett and further in view of Huang et al. (U.S. Patent No. 5,764,949, hereinafter Huang). Applicants respectfully traverse. As discussed above, the base claims 8 and 27 have been distinguished with respect to the combination of Burns and Bennett. Huang fails to provide the required disclosure and motivation that is absent from Burns and Bennett.

Huang discloses a system and method of query pass through in a heterogeneous distributed database environment that is said to allow a client to specify syntax that is only understood and processed by a database instance of a back-end server even if it is not understood by an interface module. A hybrid pass through feature provides a combination of both a pass through mode and an active mode allowing statements to be passed through to the database instance or to be processed by the interface module. To accomplish this, a pass through session is established. The scope of the pass through session is defined by statements that establish and terminate the session. Rules determine whether dynamic statements are handled in pass through mode or in active mode based on whether the statements are within or outside the scope of the pass through session. Input host variable support is provided to database instances that don't otherwise support host variables. (Abstract.)

Page 9 paragraph 29 of the Official Action cites Huang for disclosing the client sending the data access command to the data storage device over a data transmission path that bypasses the data mover computer. However, Huang is concerned with passing a database command from a client through an interface module to a server in a database management system, and not a file access command that bypasses a data mover computer of a file server programmed for managing

locks on files having data stored in data storage locations of the file server and for responding to a client request for access to a file by returning to the client the metadata of the file (including information specifying data storage locations in the file server for storing data of the file). The interface module in Huang attempts to provide clients transparent access to heterogeneous back-end database systems by translating client instructions such that they are understandable by the appropriate back-end systems. (Huang, col. 1, lines 48-59.)

It should be clear from Burns, the primary reference, that a database management system is different from a file server. In particular, the file server maps file names to data storage locations allocated for storing data of the files. The cited references fail to suggest that a file server should provide a client with information specifying the data storage locations in the file server for storing data of the file, which would be necessary for the client to access the data storage locations directly by bypassing the data mover computer in the file server. Therefore, it is not understood why a person of ordinary skill seeking to improve a file server would recognize the database system of Huang et al. as relevant to improving a file server. Furthermore, Huang teaches that statements from the client are passed through the interface module if a pass through session is established and rules determine whether dynamic statements are handled in pass through mode or in native mode. Because the statements are interpreted in the interface module, they do not bypass the interface module. Consequently, the proposed motivation for combining Burns, Bennett and Huang ("a faster and more efficient way for clients to access data inside the data storage and to reduce network traffic") is absent from the cited references themselves because network traffic is not reduced in Huang et al. Moreover, without improper hindsight, it

is not seen how any improvement in the internal operation of the interface in the database management system of Huang et al. would be applicable to improving a file server.

Hindsight reconstruction, using the applicant's specification itself as a guide, is improper because it fails to consider the subject matter of the invention "as a whole" and fails to consider the invention as of the date at which the invention was made. [T]here must be some motivation, suggestion, or teaching of the desirability of making the specific combination that was made by the applicant." In re Lee, 277 F.3d 1338, 1343, 61 U.S.P.Q.2d 1430, 1435 (Fed. Cir. 2002) (quoting In re Dance, 160 F.3d 1339, 1343, 48 U.S.P.Q.2d 1635, 1637 (Fed. Cir. 1998)). "[T]eachings of references can be combined only if there is some suggestion or incentive to do so." In re Fine, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988) (Emphasis in original) (quoting ACS Hosp. Sys., Inc. v. Montefiore Hosp., 732 F.2d 1572, 1577, 221 U.S.P.Q. 929, 933 (Fed. Cir. 1984)). "[P]articular findings must be made as to the reason the skilled artisan, with no knowledge of the claimed invention, would have selected these components for combination in the manner claimed." In re Kotzab, 217 F.3d 1365, 1371, 55 U.S.P.Q.2d 1313, 1317 (Fed. Cir. 2000). See, for example, Fromson v. Advance Offset Plate, Inc., 755 F.2d 1549, 1556, 225 U.S.P.Q. 26, 31 (Fed. Cir. 1985) (nothing of record plainly indicated that it would have been obvious to combine previously separate lithography steps into one process); In re Gordon et al., 733 F.2d 900, 902, 221 U.S.P.Q. 1125, 1127 (Fed. Cir. 1984) (mere fact that prior art could be modified by turning apparatus upside down does not make modification obvious unless prior art suggests desirability of modification); Ex Parte Kaiser, 194 U.S.P.Q. 47, 48 (PTO Bd. of Appeals 1975) (Examiner's failure to indicate anywhere in the record his reason for finding alteration of reference to be obvious militates against rejection).

With respect to claim 10, see the applicants' remarks above with respect to claim 3.

Claim 21 further distinguishes the references, in particular Burns, by reciting: "returning to said each data processing device metadata of the file including information specifying data storage locations in the data storage device for storing data of the file."

With respect to claims 22-24, see the applicants' remarks above with respect to claims 9-11.

With respect to claims 25-26, see the applicants' remarks above with respect to claims 6 and 7.

With respect to claim 28, see the remarks above for claim 21.

With respect to claims 36-41, see the remarks above for claims 21-26.

In view of the above, reconsideration is respectfully requested, and early allowance is earnestly solicited.

Respectfully submitted,

Richard C. Auchterlonie

Richard C. Auchterlonie
Reg. No. 30,607
HOWREY SIMON ARNOLD & WHITE, LLP
P.O. Box 4433
Houston, Texas 77210
(713) 787-1400

Date: 8 May 2003

HOWREY SIMON ARNOLD & WHITE HOWREY LLP WHERE LEADERS GO™

750 BERING DRIVE
HOUSTON, TX 77057-2198
PHONE: 713.787.1400 • FAX: 713.787.1440

Official

FACSIMILE COVER SHEET

DATE: 5/8/2003

TO: NAME: Commissioner for Patents, Alexandria VA 22313-1450
COMPANY: Attn: Examiner Dustin Nguyen, Art Unit 2154
FAX NO: (703) 746-7239

FROM: NAME: Richard C. Auchterlonie, Reg. 30,607
PHONE NO.: 713-787-1698 USER ID: 5137

NUMBER OF PAGES, INCLUDING COVER: 20 CHARGE NUMBER: 10830.0039.NPUS00

☐ ORIGINAL WILL FOLLOW VIA:

☐ REGULAR MAIL ☐ OVERNIGHT DELIVERY ☐ HAND DELIVERY ☐
OTHER:

☒ ORIGINAL WILL NOT FOLLOW

SUPPLEMENTAL MESSAGE:

Following is a Reply to the Official Action dated 03/17/03 in Vahalia et al., Ser. 09/261,621

CERTIFICATE OF TRANSMISSION
37 C.F.R 1.8

I HEREBY CERTIFY THAT THIS CORRESPONDENCE IS BEING FACSIMILE TRANSMITTED TO THE UNITED STATES PATENT AND TRADEMARK OFFICE, FAX NO. (703) 746-7239 ON 8 May 2003 (DATE)

TYPED OR PRINTED NAME OF PERSON SIGNING THIS CERTIFICATE:

RICHARD C. AUCHTERLONIE, REG. 30,607

8 May 2003 Richard C. Auchterlonie
DATE SIGNATURE

IF THERE ARE ANY QUESTIONS OR PROBLEMS WITH THE TRANSMISSION OF THIS FACSIMILE, PLEASE CALL 713.787.1698.